

FICHE D'INVESTIGATION DE FONCTIONNALITÉS

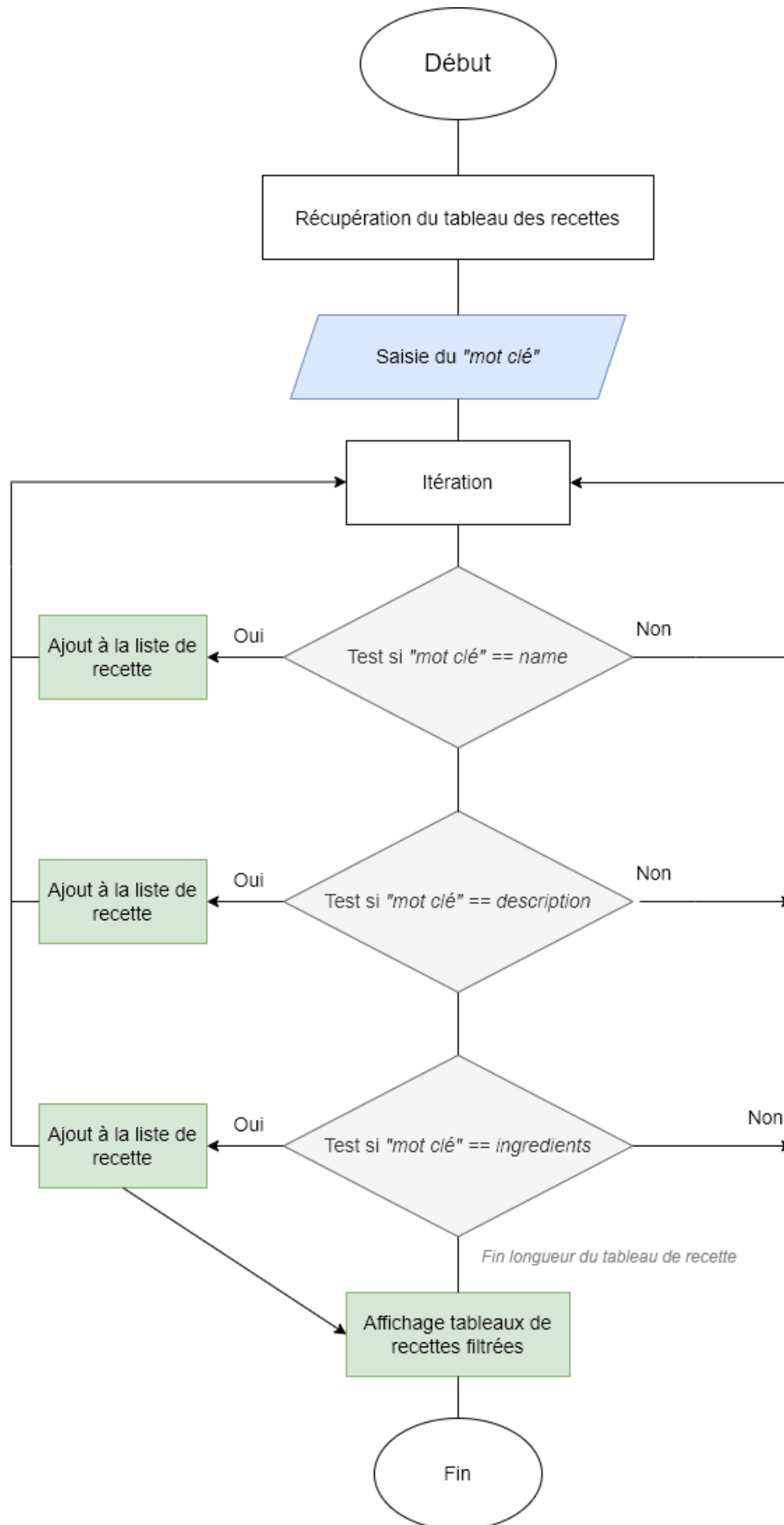
Fonctionnalité : algo de recherche	
Problématique: Un moteur de recherche fluide et performant. Les utilisateurs veulent une recherche rapide, presque instantanée	

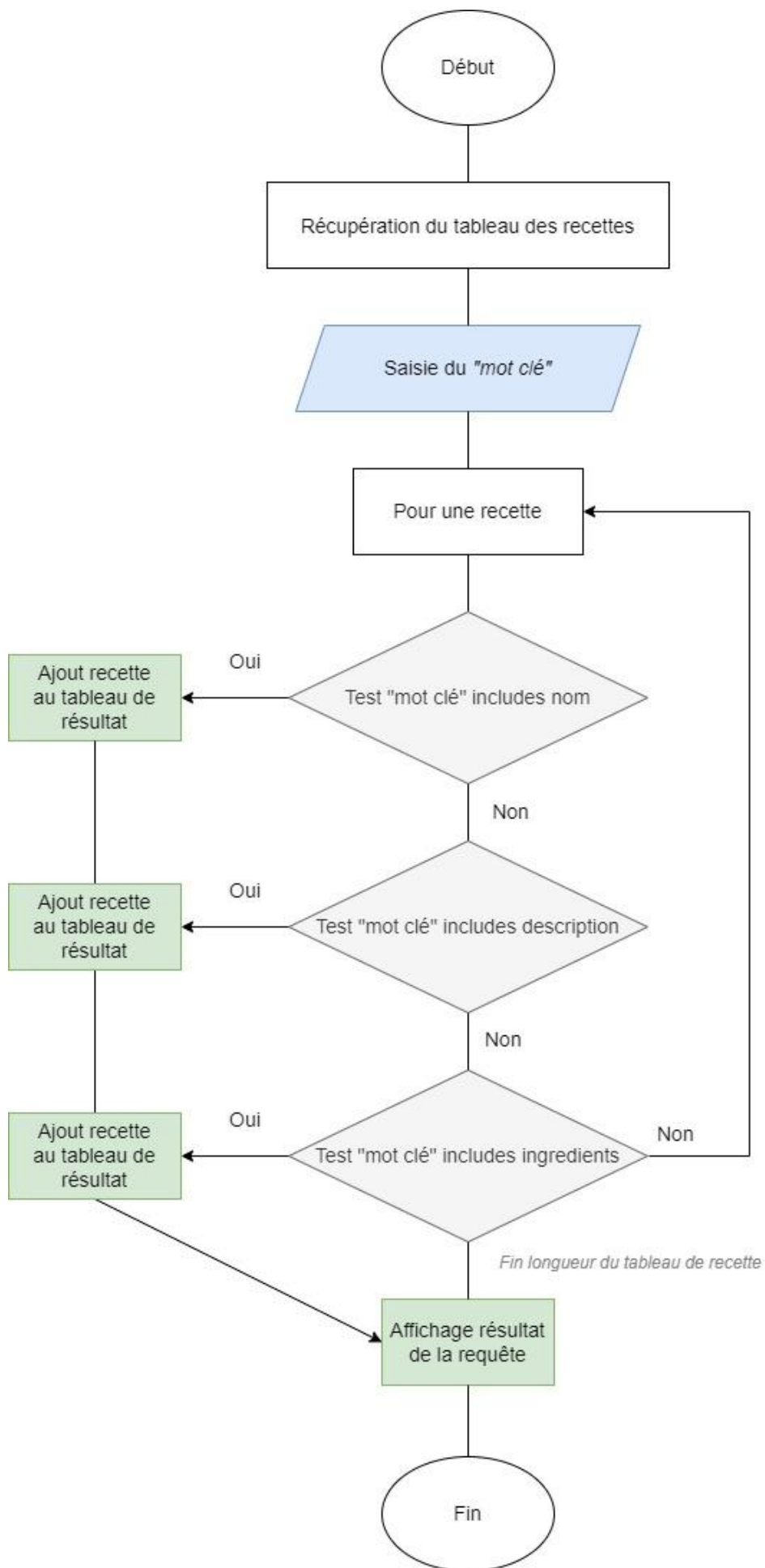
Option 1: Boucle natives for La saisie dans le moteur de recherche principal, permet de chercher dans les noms, ingrédients, description des recettes et d'afficher la recette lorsqu'il y a une correspondance.	
Avantages <ul style="list-style-type: none">- Permet de faire des itérations, à chaque passage possibilité de donner une instruction- Répète des instructions tant que la condition donnée est remplie	Inconvénients <ul style="list-style-type: none">- Complexe à écrire selon les différents niveaux de tableaux à parcourir- Utilise beaucoup de ressources
Se déclenche à partir d'une requête minimale de 3 lettres et parcourt l'ensemble du fichier json à chaque itération.	

Option 2 : Méthodes de l'objet <i>Array</i> - filter + map + foreach La saisie dans le moteur de recherche principal, permet de chercher dans les noms, ingrédients, description des recettes et d'afficher la recette lorsqu'il y a une correspondance.	
Avantages <ul style="list-style-type: none">- Code lisible, nomenclature et syntaxe claire- Exécution rapide, utilise moins de ressources que la boucle for- Stock dans un nouveau tableau les valeurs qui ont répondu à la condition	
Déclenchement de la recherche à partir de 3 caractères saisis, et filtre le fichier Json	

Solution retenue : L'option 2 avec les méthodes Array semblent plus pertinentes du point de vue de la maintenabilité, la lisibilité et la compréhension du code. D'autres part le benchmark d'exécution des deux script sur un échantillon de 1, 10, puis 50 recettes, montrent des performances sans appel des méthodes array, par rapport à la boucle for. '(Voir annexe)
--

BOUCLES FOR

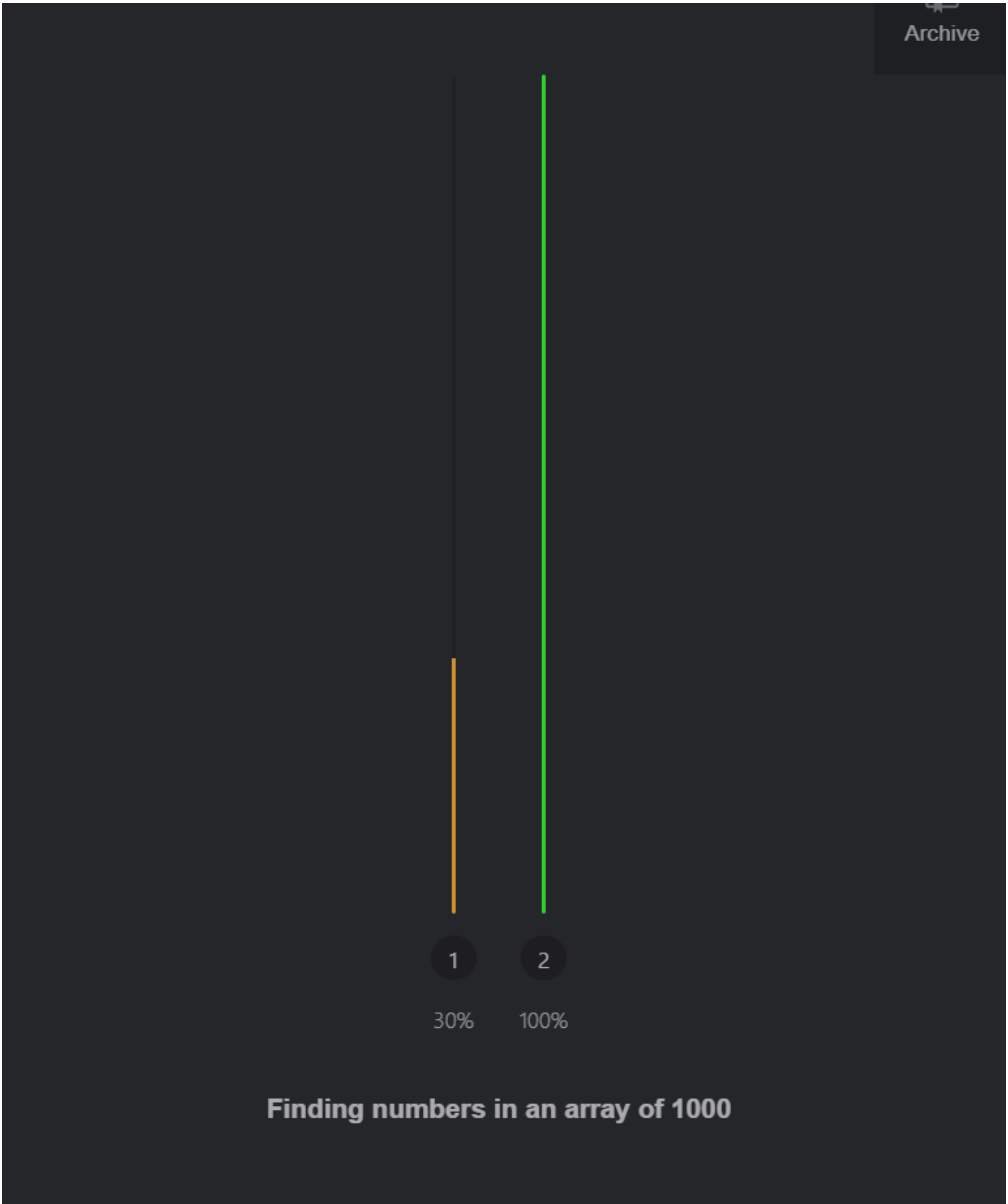




PERFORMANCE AU BANC D'ESSAI *PERF.LINK*

CAS1

Boucle for (1)		Array.filter Array.map (2)
1	Nombre recettes	1
'coco'	Mots clés	'coco'
146 440 ops/s	Nombre opération /sec	555 710 ops/s
3	Nbre de test	3
30%	Score final le plus haut	100%



CAS 2

Boucle for (1)		Array.filter Array.map (2)
10	Nombre recettes	10
'glaçons'	Mots clés	'glaçons'
14 690 ops/s	Nombre opération /sec	16 130 ops/s
3	Nbre de test	3
91%	Score final le plus haut	100%



CAS 3

Boucle for (1)		Array.filter Array.map (2)
50	Nombre recettes	50
'poisson'	Mots clés	'poisson'
1877 ops/s	Nombre opération /sec	5340 ops/s
10	Nbre de test	10
35%	Score final le plus haut	100%

